REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 11, 2005	**Final Report, 15 Nov.01- 14 Nov.04**

4. TITLE AND SUBTITLE
REAL-TIME IMPLEMENTATION OF A FUZZY LOGIC CONTROLLER FOR DC-DC SWITCHING CONVERTERS

5. FUNDING NUMBERS
DAAD19-02-1-0006

6. AUTHOR(S)
Ahmed Rubaai, PI

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Howard University, 2400 6th Street, Northwest, Washington, DC 20059

8. PERFORMING ORGANIZATION REPORT NUMBER
ORA-01-163-G (FRS 634715)

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U. S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
42997.1-EL-H

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This report presents a successful implementation of a fuzzy logic controller structure for dc-dc switching converters and evaluates experimentally its sensitivity for variable supply voltages and load resistance variations. The optimum topology of the controller structure is determined using experimental tests. An advanced test bed system is used to evaluate the robustness capacities of the controller under varying loading conditions and input voltage variations. The experiment is performed using a low cost micro-controller PIC16F877 in order to verify the design performance over a wide range of operating conditions. Experimental results are obtained using appropriate scaling factors associated with the input variables of the fuzzy controller. The controller show very interesting tracking features and is able to cope with load changes and input voltage variations. The proposed controller structure is general and can be directly applied to any dc-dc converter topologies. The fuzzy controller structure is compared experimentally with the existing PI control used in industry. It yields a better dynamic performance without overshoot.

14. SUBJECT TERMS
Fuzzy-Logic Controller, DC-DC Converters, Laboratory Setup, Test results

15. NUMBER OF PAGES
22

16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION ON THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	**UNCLASSIFIED**	**UNCLASSIFIED**	**UL**

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. 239-18
298-102

TABLE OF CONTENTS

LIST OF FIGURES

REPORT SUMMARY

This report presents a successful implementation of a fuzzy logic controller structure for dc-dc switching converters and evaluates experimentally its sensitivity for variable supply voltages and load resistance variations. The optimum topology of the controller structure is determined using experimental tests. An advanced test bed system is used to evaluate the robustness capacities of the controller under varying loading conditions and input voltage variations. The experiment is performed using a low cost micro-controller PIC16F877 in order to verify the design performance over a wide range of operating conditions. Experimental results are obtained using appropriate scaling factors associated with the input variables of the fuzzy controller. The controller show very interesting tracking features and is able to cope with load changes and input voltage variations. The proposed controller structure is general and can be directly applied to any dc-dc converter topologies. The fuzzy controller structure is compared experimentally with the existing PI control used in industry. It yields a better dynamic performance without overshoot.

I. INTRODUCTION

Power conversion circuits are essentially non-linear control devices and are often classified in four categories: 1) ac-ac converters, 2) ac-dc converters, 3) dc-ac converters, and 4) and dc-dc converters. Choppers themselves are generally divided into two groups: step-down or buck converter and step-up or boost converter [1]. For buck converters with constant output voltage, it is always desirable that the output voltage remains unchanged in both steady state and transient operations whenever the supply voltage and/or load current are disturbed. This condition is known as zero-voltage regulation and it means that the output voltage is independent of the supply voltage and the load current. To achieve zero-voltage regulation, the choice of the control method plays a very critical role in the performance of converters. Using an equivalent small-signal linear model of the converters, conventional frequency domain analog methods are predominantly used in designing the controllers. In conventional methods, performance is traded off for simplicity since for nonlinear systems the small signal linear models have a rather restricted validity. The most commonly used control method in converters is the direct duty ratio control [2-5]. In this control method the output voltage is constantly monitored, fed-back, and compared with a reference voltage and their difference (the error) is amplified and used to control the duty ratio of the converter in such a way that the output voltage remains constant. This approach, however, cannot eliminate the supply voltage and load current disturbances until they are detected at the output. A method to achieve zero-voltage regulation in buck converters has been introduced in [4]. Using circuit analysis, a direct relationship between the average output voltage and the reference voltage is determined. Based on this relationship, a proper control law (Function Control) is developed. Employing an averaged low frequency linear topology of buck converters, the control law shows that the output voltage is independent of both the input voltage and the load current and, thus, a zero-voltage regulation can be achieved. Using function control, however, the exact relationship between the input and output voltages becomes too complex to be practically executed. Another popular method is the current mode control, where the inductor current is also monitored and fed-back together with the output voltage. By a proper design, current mode control can

eliminate the input voltage disturbances but it cannot eliminate the load current disturbance [6]. Feed-forward types of controllers have also been designed by sensing the input voltage to improve line regulation in applications with a wide range of input voltages and load currents. However, direct sensing of the input voltage through a feed-forward loop may induce large-signal disturbances that could upset the normal duty-cycle of the converter.

Using human linguistic terms and common sense, several fuzzy logic-based controllers have been developed in [8-15]. These fuzzy controllers have shown promise in dealing with nonlinear systems and achieving voltage-regulation in buck converters [10-12]. In all publications, the closed-loop control scheme entails in incorporating engineering knowledge into the automatic control system by using the intuition and experience of the designer. Nevertheless, the main problem with fuzzy logic is that there is no systematic procedure for the design of a fuzzy controller.

In this report, a closed-loop control structure incorporating fuzzy logic with a small rule base and an effective realization has been implemented for a class of hard- switching dc-dc converters. The fuzzy logic controller structure incorporates attractive features such as simplicity, good performance, and automation, while using a low-cost hardware and software implementation. Two categories of tests that cover the two basic performance areas, load regulation and line regulation, are carried out to evaluate the performance of the controller. Experimental results show that the effective smooth output voltage control and tracking of switching dc-dc converters can be achieved by the proposed controller structure, thus making the fuzzy logic control suitable for any type of dc-dc converter topologies.

II. FUZZY CONTROLLER STRUCTURE

The structure built-up here is a two-input single-output controller. The inputs are the variable voltage error, e(k) and the change in voltage error, Δe(k). Consequently, the input to the dc-dc converter would be the duty cycle that is actually the output of the fuzzy controller. The operational structure of the proposed fuzzy controller is shown in Fig. 1. It consists of three building blocks: 1) a fuzzification block that expresses quantitative action to qualitative action, 2) Fuzzy inference engine that generates the fuzzy rules, and 3) a defuzzification block that articulates qualitative action to quantitative action. Fuzzification translates a numeric value for the error e(k) or change in error Δe(k) into a linguistic value such as big or small with a membership grade. Defuzzification takes the fuzzy output of the rules and generates a "crisp" numeric value used as the control input to the dc-dc converter. The controller qualitatively captures the dynamics of the dc-dc converter and executes this qualitative idea in a real-time situation.

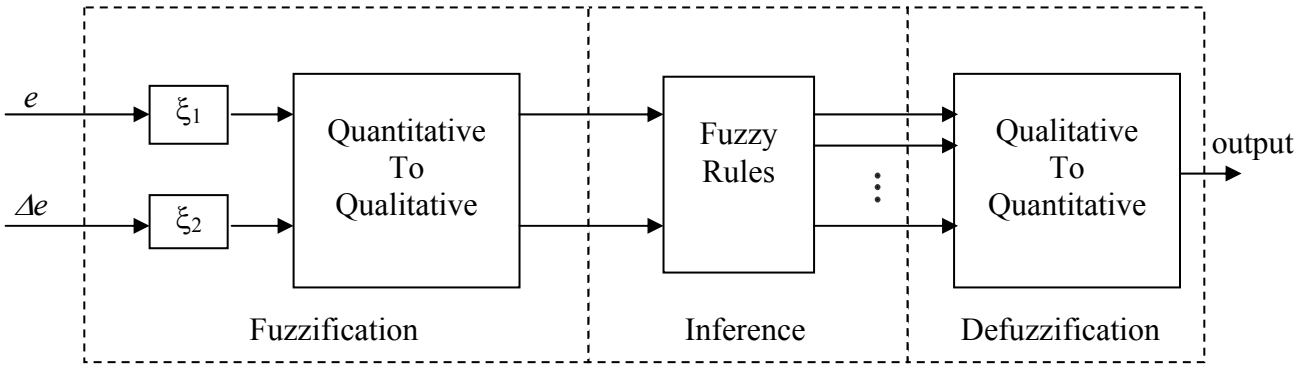

Fig. 1 Fuzzy Controller Structure

The overall architecture of a dc-dc converter topology with the fuzzy controller structure is given in Fig. 2. The converter is represented by a "black box" from which we only extract the terminals corresponding to input voltage (V_i), output voltage (V_o), and controlled switch (S). From these measurements, the controller provides a percentage duty cycle signal for a peripheral interface microcontroller PIC16F877 which generates the converter actual duty cycle.

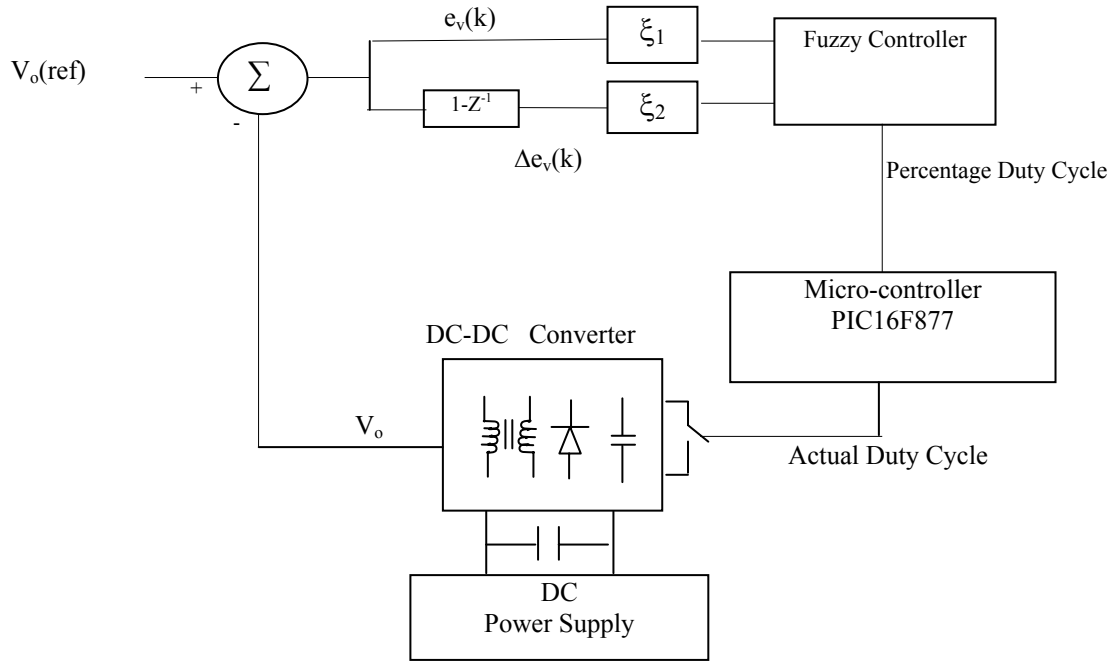

Fig. 2 Fuzzy Controller-based converter topology

III. OPTIMUM TOPOLOGY OF THE FUZZY CONTROLLER STRUCTURE

The main problem with fuzzy controller generation is related to the definition of its optimum topology such as scaling factors, number of fuzzy sets, shape of the fuzzy sets, and decision making. This section is devoted to the topology determination of the proposed fuzzy controller structure. In order to define the optimum topology, the number of fuzzy sets for the fuzzy controller structure has been determined experimentally using the measured responses of the dc-dc converter. The actual responses have been compared for three, four, five and seven fuzzy sets. It has been found that the use of more than five linguistic sets does not improve the tracking accuracy of the dc-dc converters, but in fact increases the computation time. Consequently, four linguistic sets: positive large (PL), positive small (PS), negative large (NL), and negative small (NL) have been chosen for the input variable voltage error, e(k), and three linguistic sets: positive medium (PM), zero (ZE), and negative medium (NM) have been chosen for that for the change in voltage error, Δe(k). Additionally, five linguistic sets: positive large (PL), positive small (PS), Zero, negative large (NL), and negative small (NL) have been chosen for the output variable of the fuzzy controller structure. The fuzzy controller utilizes triangular membership functions on the controller input. The triangular membership function is chosen

owing to its simplicity. For the change in voltage error, Δe(k), the initial values of the premise parameters (the corner coordinates of the triangle) are chosen so that the membership functions are equally spaced along the operating range of each input variable.

The scaled input and output membership functions sets are shown in Figs. 3 through 5. The actual error measured from the feedback network ranges from 0.8V to about 6.0V while the control signal ranges from 16% to 50% duty cycle.

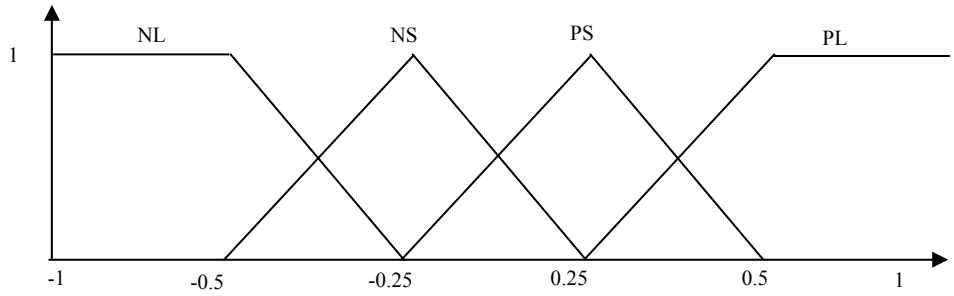

Fig. 3 Input membership functions for the error, e (k)

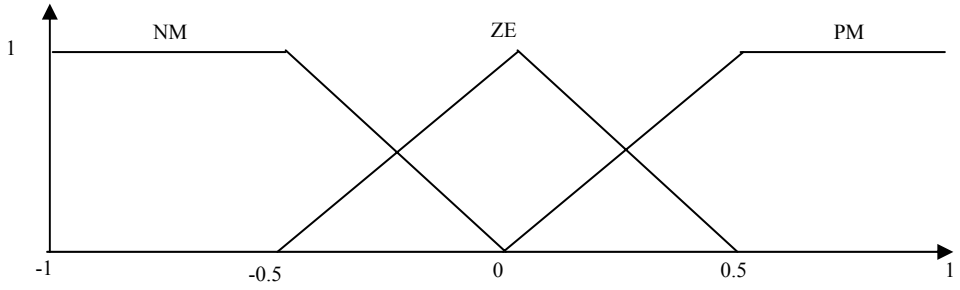

Fig. 4 Input membership functions for the change in error, Δe(k)

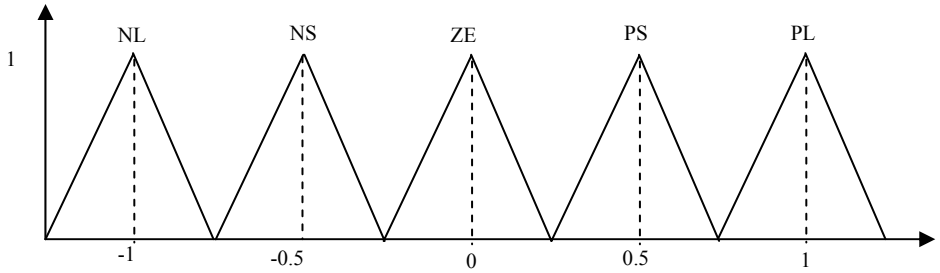

Fig. 5 Output membership functions for the control

The scaling factors ξ_1 and ξ_2 are determined using experimental tests in such a way that the normalized inputs e(k) and Δe(k) are well adapted to the universe of discourse [-2, 2] for any operating point.

The linguistic control rules are established considering the dynamic behavior of the dc-dc converter as well as analyzing the error and its variation. These control rules can be expressed as

IF e(k) is PL AND Δe(k) is PM THEN O is PM OR

IF e(k) is NL AND Δe(k) is NM THEN O is NM OR **…**

The symbol "IF" of the rules is called the premise, while the symbol "THEN" is the consequence. The "AND" operator is used to link the premises and the "OR" operator to link the rules. To obtain control decision, the max-min inference method is used. It is based on the minimum function to describe the "AND" operator present in each control rule and the maximum function to describe the "OR" operator.

The output of the fuzzy controller structure is crisp, and thus a combined output fuzzy set must be defuzzified. To express the qualitative action in a quantitative action, the sum-product composition method was used [20]. It calculates the crisp output as the weighted average of the centroids of all output membership functions. The output of the controller can be expressed as

$$O_o = \frac{\left(\sum_m O_m * a_{Cm} * b_{Cm}\right)}{\sum_m (O_m * b_{Cm})}$$

where a_{Cm} and b_{Cm} are the centers and widths of the output fuzzy sets respectively, for $m = 1,...5$. The values of the $b_{Cm}'s$ were chosen to be unity. This scaled output corresponds to the control signal (percent duty cycle) to be applied to maintain the output voltage at a constant value. Fig. 6 represents the normalized output O_o of the proposed fuzzy controller structure as a function of the error e(k) and the change of error Δe(k). Clearly Fig. 6 illustrates the nonlinear characteristics of the proposed fuzzy controller structure.

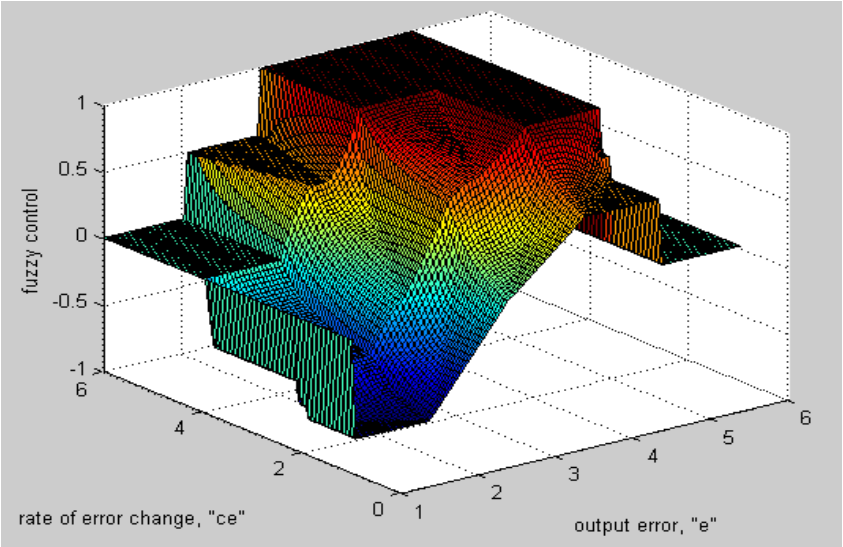

Fig. 6 Fuzzy control surface

IV. DESCRIPTION OF THE DC-DC CONVERTER SYSTEM

The DC-DC converter system utilized in this project is composed of six distinct circuit networks working together to form a complete DC-DC dual-output switch-mode power regulator. It uses the method of integrated-magnetics. The six distinct circuit networks are:

- the bias voltage regulator network,
- the 100 KHZ clock,
- the pulse-width-modulator (PWM) & ramp generator subsystem,
- the power switch driver interface,
- the feedback error amplifier & ground isolation network,
- the over-current sensor and the integrated-magnetic (IM) power stage.

Fig. 7 shows a block diagram of the actual dc-dc converter system before it was modified.

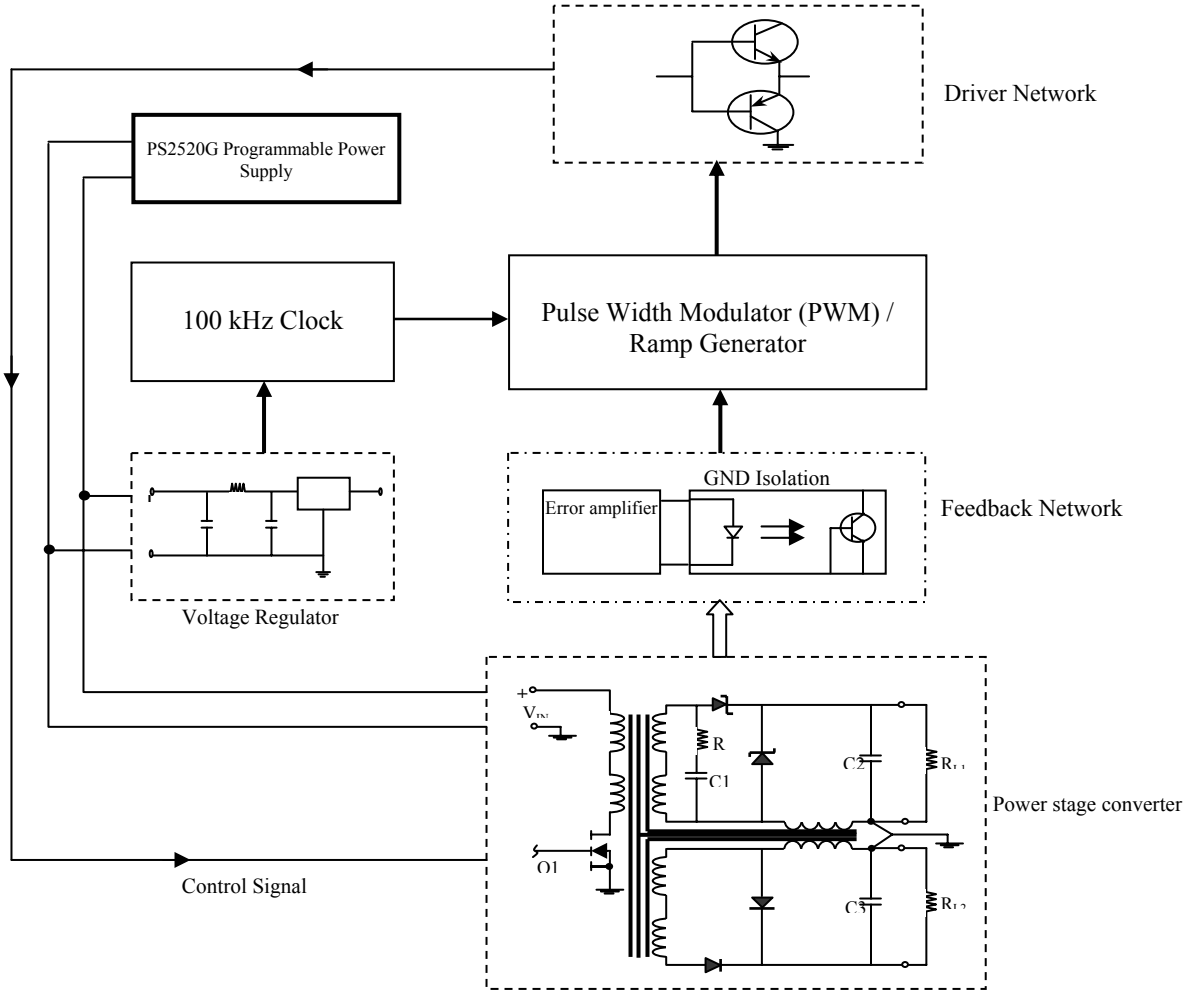

Fig.7 Block diagram of converter system before modification

Bias voltage regulator

The purpose of the bias voltage regulator circuit is to provide a source of regulated 12VDC power for the clock, PWM and driver circuits of the converter. A voltage regulator circuit is shown in Fig. 8. The resistor and capacitor C2 form a low-pass noise filter to remove high-frequency noise at the input.

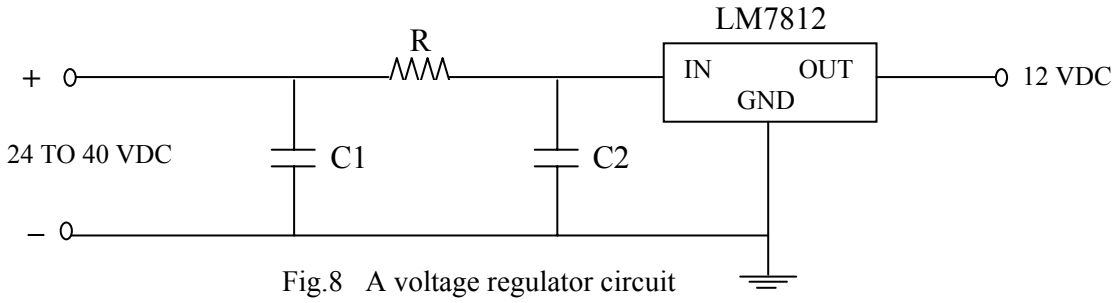

Fig.8 A voltage regulator circuit

100 khz clock

The converter system performs its power translation at a fixed-frequency rate of nominally 100 khz. The purpose of the clock network is to provide a dynamic voltage set at this frequency for use by the PWM controls of the system. The output falling edge of the clock used as a "trigger" for the PWM controls

PWM & RAMP generator

The heart of the control system of the dc-dc converter system is its pulse-width-modulation (PWM) network. Any errors detected in the output voltage of the converter system by the feedback controls are used to automatically adjust the ON time of the power stage to maintain the two outputs of the converter at near-constant 5VDC and 15V levels as the input voltage to the converter and the output loads are changed. These voltage adjustments are made at a fixed-frequency rate of' 100 khz developed by the clock circuit. It is built using a LM555 timer, resistors, capacitors and a transistor.

Driver

The power switch Q1 is operated through a low-impedance turn-on and turn-off signal provided by a buffer circuit formed around two transistors. During ON times of the power switch Q1, the drain-to-source current depends on the load being drawn from the converter system.

Error amplifier and ground isolation elements

The role of the feedback control is to provide to the PWM network the value of the error at the output, for this latter to reacts in a way to bring the output voltages to the constant desired values. The feedback network is built around an opto-coupler which provides ground isolation between the input and the output. A potentiometer allows the adjustment of the two output voltages to desired levels.

Power stage and output filters

The heart of the converter system is the DC-DC switch-mode power stage, where both transformer and inductive functions have been combined into a single magnetic assembly.

The power stage concept is based on that of a “dual-output forward" configuration operating in a continuous mode of energy storage. This is shown in Fig. 9. The energy is transferred from the input

power source (VIN) to the two secondary sides of the transformer when the power MOSFET switch Q1 is turned ON. The on time is a fraction of the clock period. It has to be short and long enough to bring the output to a level above the desired voltage (5V and 15V). When the switch turns off, the output stored energy, located in two inductors included in the transformer, will then start discharging causing the voltages to drop. As soon as the voltages go below the desired values, another impulsion will be applied on Q1 (Q1 ON), to bring the voltage back upwards.

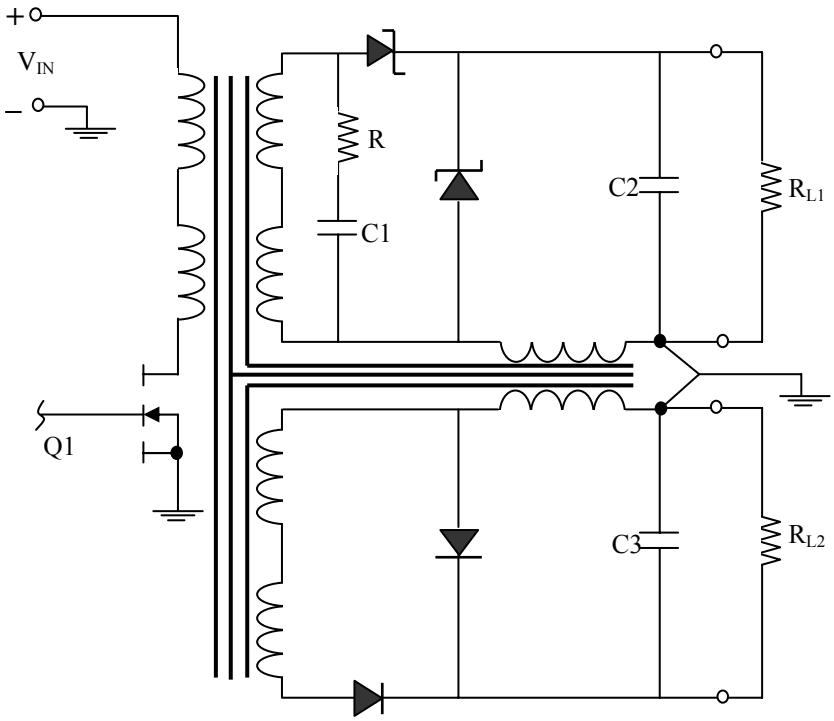

Fig.9 The power stage of the converter system

Test Results

Various test cases were done on the dc-dc converter system to help get a better understanding of its operations. A summary of these tests is shown in Fig. 10 which shows the optimum trajectories of the Duty Cycle, the Output Voltage, and the Input Current with variations in the input voltage. This test results was very significant in the design of the rules of the Fuzzy logic controller. For clarity reasons, the variation in the input current was redrawn in Fig. 11 since it was not very clear in the Fig. 10.

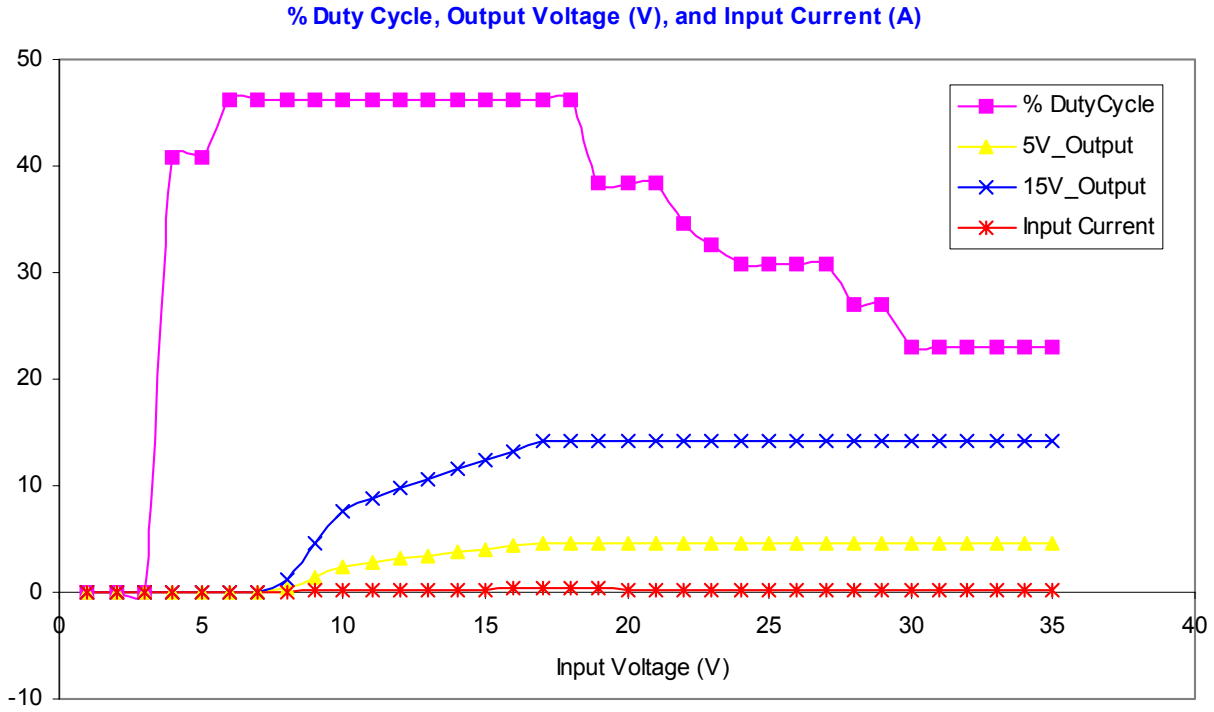

Fig. 10 Optimum trajectories of the % Duty Cycle, the output Voltage, and the Input Current with variations in the input voltage

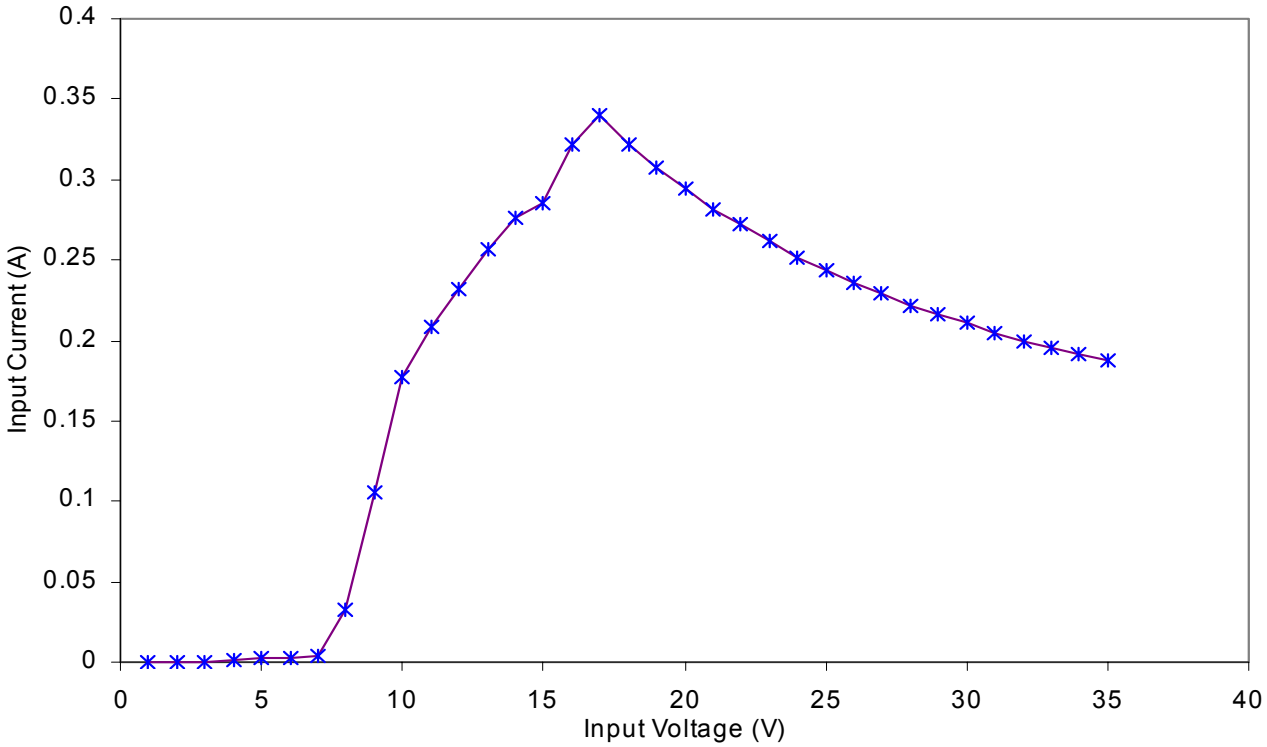

Fig. 11 Trajectory of the Input Current with variations in the input voltage

V. EXPERIMENTAL IMPLEMENTATION

Fig. 12 shows the configuration of the experimental implementation which consists of: 1) a DC-DC switch-mode power stage converter 2) a 14-bit PCI Data Acquisition Processor (DAP 840/103) [16], 3) a termination board (MSTB 010-06-C1Z) [16], 4) a Pentium III 550-MHz personal computer (PC) with Windows NT 4.0, and 5) a micro-controller (PIC16F877) [17]. A snapshot of the laboratory experiment is displayed in Fig. 13. The power stage concept is based on that of a "dual-output forward" configuration operating in a continuous mode of energy storage [18]. When the power MOSFET switch Q1 is turned ON energy is transferred from the input power source (VIN) to the two secondary sides of the transformer. The voltage potential across the terminals of C11 will be that of the reflected line voltage, namely $N_{S1}*V_{IN}/N_P$. When Q1 turns OFF, the 5V and 15V load power is sustained by the energies of the two 'inductors'(L1 & L2) and the energy stored in C11 as a result of the potential $N_{S1}*V_{IN}/N_P$, will then flow back into the 5V secondary winding in a resonant manner with the magnetization inductance of the transformer. This causes the voltage across the 5V secondary to reverse polarity in a sinusoidal manner, until the energy in C11 is completely dissipated. The PCI Data Acquisition Processor (DAP 840/103) occupies one expansion slot in the Personal computer and has onboard processor, (TI486SXLC2-50 CPU), 14-bit A/D converter, 50ns TIME resolution, 800K samples per second, memory, and a dedicated multitasking real-time operating system. The MSTB (010-06-C1Z) termination board allows secure connection of discrete wires to the DAP 840/103 and it combines analog and digital termination on the same board.

The feedback network provides as input to the fuzzy controller the error value at the output, for the appropriate control signal to be generated. It is built around an opto-coupler that provides ground isolation between the input and the output with a potentiometer for the adjustment of the two output voltages to desired levels. The termination board, DAP 840/103, reads the error value from the feedback network using an input channel pipe to the PC in binary format. The source program (written

in Matlab and Dapview language) running on the PC is configured to read the data and correctly processed.

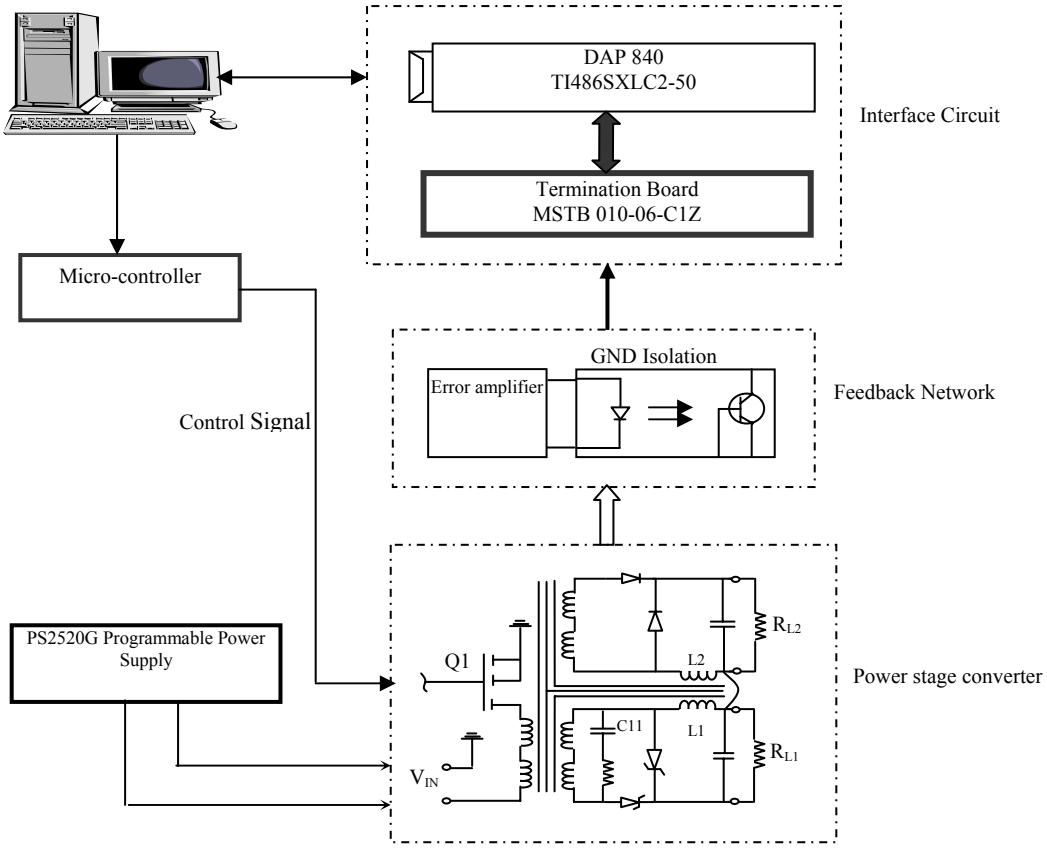

Fig.12 Experimental setup after modification of the original converter

The processed data is finally sent as an input to the fuzzy adaptive controller code running on the PC to issue the appropriate command signal for the microcontroller through pin 3 of the RS232 to generate the control signal, which is the duty cycle. Using a microcontroller the duty cycle is generated by a peripheral Interface Controller (PIC16F877), which uses the Harvard Architecture and mostly used in RISC (Reduced Instruction Set Computer) Computers. It has a separate program bus and data bus, which can be of different widths. A single instruction cycle time of the PIC 16F877 is 0.2 μs. A code was written using MPLAB and loaded into the PIC16F877 to generate pulses at 100 kHz with variable duty cycle depending on the input data received through pin 3 of the RS232 sent by the fuzzy logic controller running on the PC. The output pulses are sent through pin 2 of the RS232 to the gate of transistor Q1 which in turn produces the necessary drive pulse to the power stage of the converter to keep the output voltage constant.

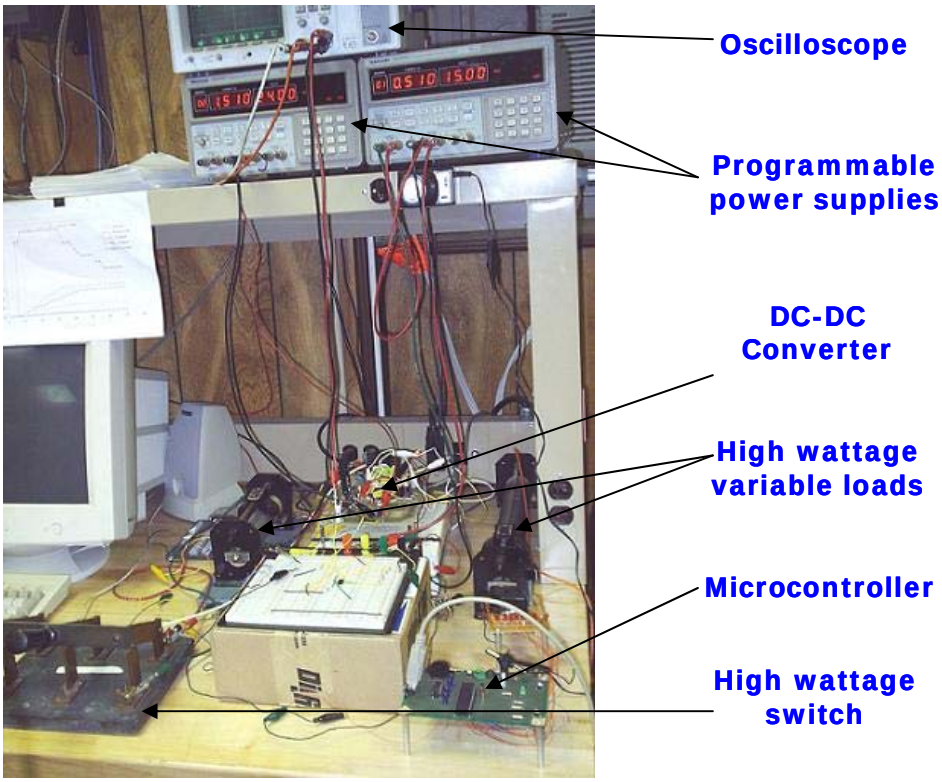

Fig.13 Snapshot of the laboratory setup

Test Measurements

Various test measurements were done on the DC-DC converter after it has been modified to help in designing the intelligent controller. Fig. 14 shows a test case which shows how the output voltage changes with various control signals (duty cycle). A percentage duty cycle was maintained throughout the input voltage variations to see the effect on the output voltage. This was repeatedly done for 16%, 24%, 32%, and 40% duty cycles.

In Fig. 15 it shows the error voltage variations for the from the feedback network for various control signals. Each of the control signals (duty cycle) was applied separately to the DC-DC converter while the feedback network is being monitored. It shows that the feedback network goes into operation at different voltage levels based on the control signal being applied. This fact was also utilized in the design of the fuzzy logic controller.

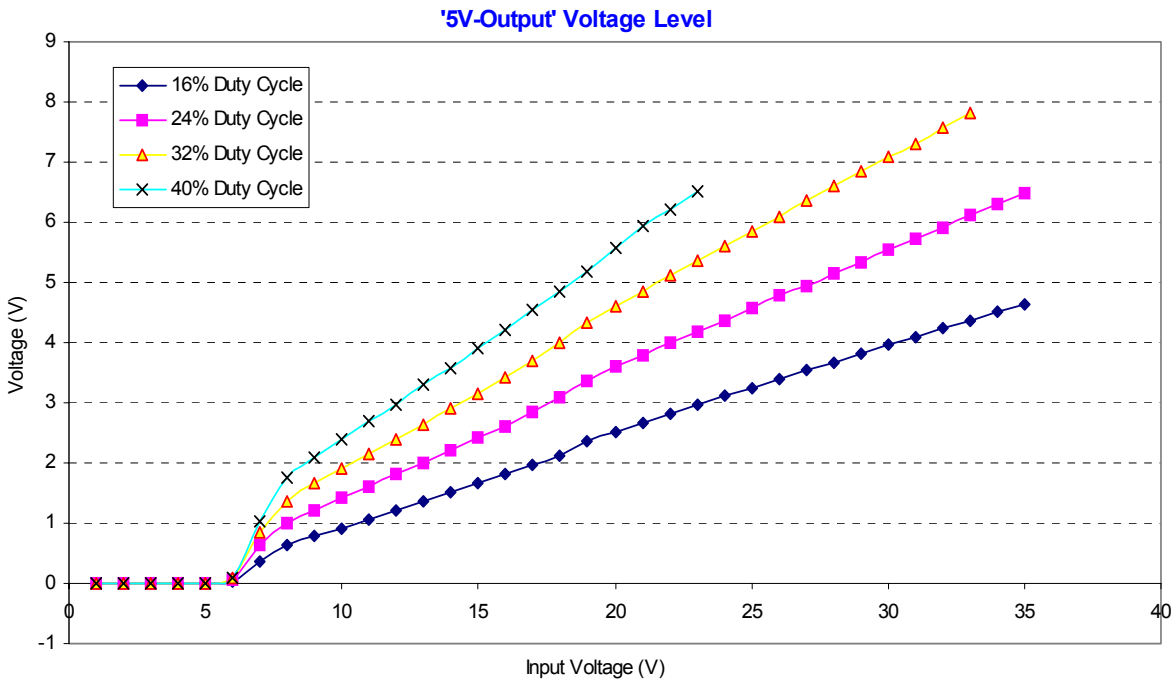

Fig.14 Variation of output voltage with % Duty Cycle

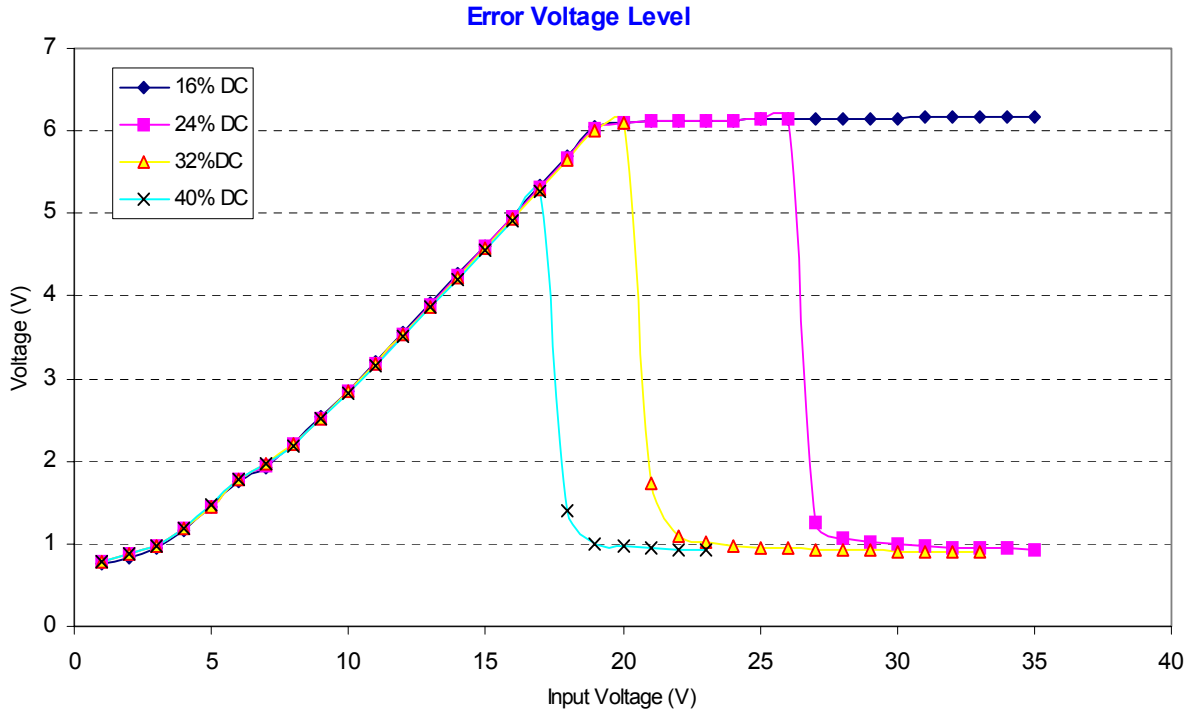

Fig.15 Variation of feedback error voltage with % Duty Cycle

VI. DISCUSSION OF EXPERIMENTAL RESULTS

Several test cases were conducted to assess the performance of the proposed fuzzy control system. However, for briefness, only few cases are reported for illustration purposes. Selected cases of test results performed on switch mode power stage DC-DC converter are illustrated in Figs. 16-21.

Fig. 16 shows the output voltage response of the DC-DC converter with a step input voltage variations from 22V-to-24V and the back to 22V to complete a cycle. This was made possible using the programmable power supply (PS2520G) from Tektronix. In Fig. 17, a step input voltage variations from 19V-to-24V-to-31V and the back to 24V and 19V to complete a cycle was used. In the above two case studies, it will be observed that, the proposed fuzzy controller maintains the output voltage at the desired 5V with slight overshoot during each voltage change. One of the scenarios was investigated without the controller and the result is shown in Fig. 18. Without the fuzzy control it was realized that the output voltage response of the converter follows the step pattern of the input voltage and goes beyond 8V when the input voltage is 31V during its cycle.

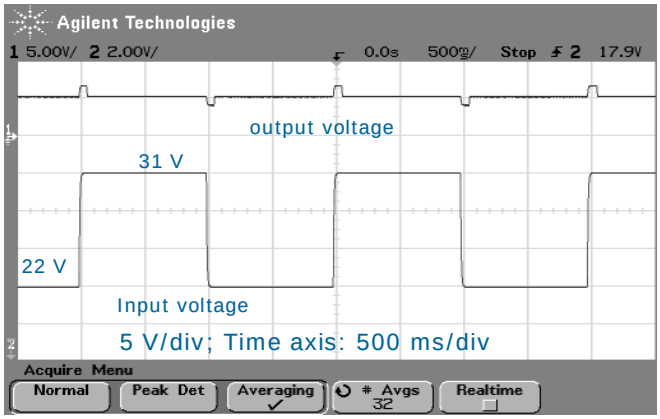

Fig.16 Output voltage response with fuzzy control to step input voltage variations from 22V and 31V

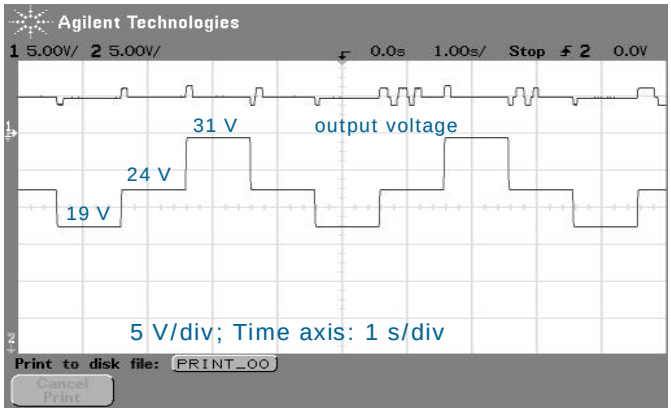

Fig.17 Output voltage response with fuzzy control to step input voltage variations from 19V, 24V and 31V

Fig. 19 shows the output voltage response of the DC-DC converter with a step load variations from 10 ohms to 2 ohms. Fig. 20 shows the response of the DC-DC converter to load transient from 10 ohms to 2 ohms (equivalent output current was about 3A) for about 0.8 of a second. Again, the control action of the fuzzy system was able to maintain the output voltage at the desired 5V with slight overshoot in

both cases. Fig. 21 show a screen capture of the output voltage response together with the control signal (duty cycle) in the micro-second time division range for a 40% duty cycle.

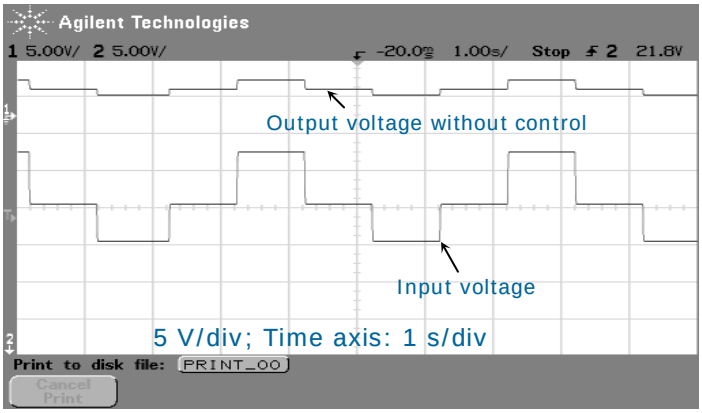

Fig.18 Output voltage response without control to step input voltage variations from 19V, 24V and 31V

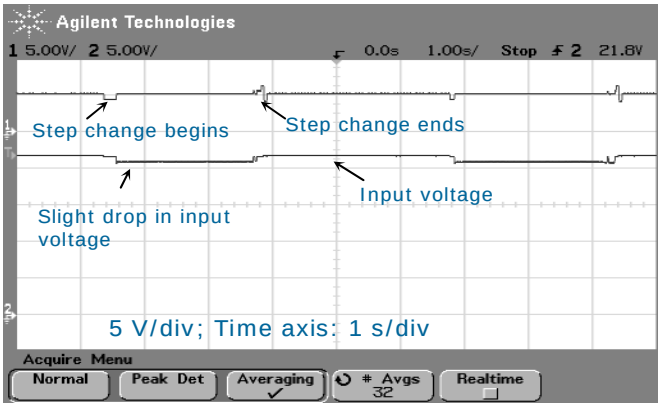

Fig.19 Output voltage response to a load step change from the normal 10 ohms to 2 ohms (equivalent to 3 A output current)

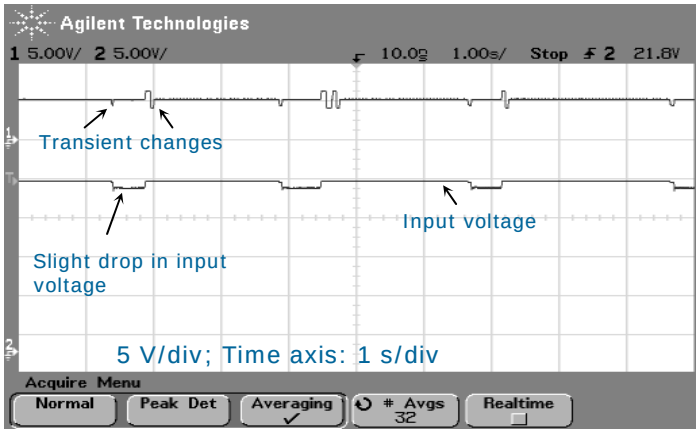

Fig.20 Output voltage response to a load transient from the normal 10 ohms to 2 ohms (equivalent to 3 A output current)

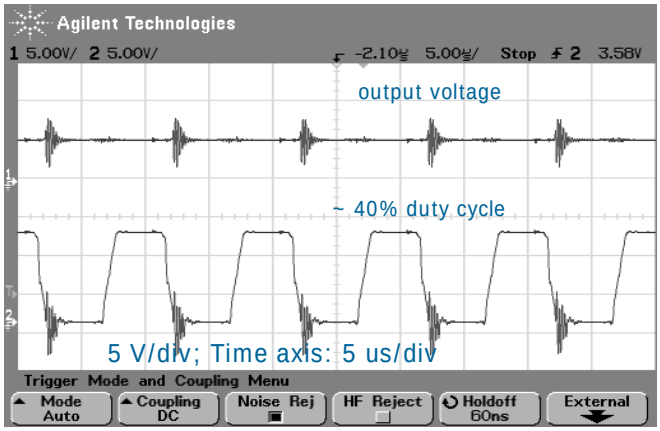

Fig.21 Output voltage response in the micro-second range for an approximately 40% duty cycle control signal

VII. CONCLUSIONS

A low cost fuzzy logic controller structure using a small number of rules has been implemented experimentally for a special class of hard-switching dc-dc converters. Many test cases demonstrate that the fuzzy controller structure is capable in reducing the effect of different disturbances such as load changes and input voltage variations commonly found in industry. Experimental results show the ease of applying fuzzy control to dc-dc converters, as an interesting alternative to existing conventional industrial controllers. Also, test results illustrate that the proposed fuzzy logic controller structure can provide considerable control performance over a wide range of operating conditions. Fuzzy logic appears to be a valid element for generalization to many control applications.

REFERENCES

[1] J. Foutz, "Switching-Mode Power Supply Design," Internet Site:http://www.smpstech.com/tutorial/tO2top.htm, August 1999.

[2] J. Sun and H. Grotstollen, "Symbolic analysis methods for averaged modeling of switching power converters," IEEE Trans. Power Electronics, vol. 12, pp. 537-546, May 1997.

[3] H. Sira-Ramirez, "Design of P-I controllers for DC-to-DC power supplies via extended linearization," Int. J. Control, vol. 51, no. 3, pp. 601-620, 1990.

[4] Kugi and K. Schlacher, "Nonlinear H_∞-controller design for a DC-to-DC power converter," *IEEE Trans.* Contr. System Technology, vol. 7, pp. 230-237, Mar. 1999.

[5] G. Escobar, R. Ortega, H. Sira-Ramirez, J.P. Vilain and I. Zein, "An experimental comparison of several nonlinear controllers for power converters," IEEE Control Systems, vol. 19, no.1, pp. 66-82, Febr. 1999.

[6] Y. Liu and P. C. Sen, "A Novel Method to Achieve Zero-Voltage Regulation in Buck Converter," IEEE Trans. Power Electronics, Vol. 10, No. 3, pp. 171-180, May 1995

[7] R. Redl, and N. 0. Sokal, "Current-mode Control, Five Different Types With the Three Basic Classes of Power Converters: Small-Signal AC and Large-Signal DC Characterization, Stability Requirements and Implementation of Practical Circuits," IEEE PESC Conference, pp. 771-785, 1985

[8] J. Yen and R. Langari, Fuzzy Logic: Intelligence, Control, and Information, New Jersey: Prentice Hall, 1999.

[9] P. Maftavelli, L. Rossetto, et al., "General-purpose Fuzzy Controller for DC-DC Converters," IEEE Trans. Power Electronics, Vol. 12, NO 1, January 1997.

[10] T Gupta and R. R. Boudreaux, "Implementation of a Fuzzy Controller for DC-DC Converters Using an Inexpensive 8-b Micro-controller," IEEE Trans. Industrial Electronics, Vol. 44, No. 5, October 1997

[11] W. So, C. K. Tse, and Y. Lee, “Development of a Fuzzy Logic Controller for DC/DC Converters: Design, Computer Simulation, and Experimental Evaluation,” IEEE Trans. Power Electronics, Vol. 11, No. 1, January 1996

[12] J. Arias, A. Arias, S. Gomariz, and F. Gumjoan, “Generating Design Rules For Buck Converter-Based Fuzzy Controllers,” Proceedings of the IEEE International Symposium on Circuits and Systems, Vol. 1, pp. 585-588, May 12-15, 1996

[13] S. Gomariz, E. Alarcon, A. Martinez, and A. Poveda, “Minimum Time Control of Buck Converter by Mean of Fuzzy Logic Approximation,” Proceedings of Industrial Electronics Conference (IECON), Vol. 2, pp. 1060-1065, Aug3l - Sep4, 1998

[14] A. Balestrino, A. Landi, and L. Sani, “CUK Converter Global Control via Fuzzy Logic and Scaling Factor,” IEEE Trans. Ind. Applicat., Vol. 38, No. 2, pp. 406-413, March/April 2002.

[15] E. Vidal, L. Martine, F. Guinjoan, J. Calvente, and S. Gomariz, “Sliding and Fuzzy Control of a Boot Converter using an 8-Bit Microcontroller,” IEE Proc., Electr. Power Appl., Vol. 151, No. 1, pp. 5-11, 2004.

[16] J. Alvarez, I. Cervantes, G. Espinosa, P. Maya, and A. Morales,” A Stable design of PI Control for DC-DC Converters with an RHS zero,” IEEE Trans. Circuits and Syst., Vol. 48, No. 1, pp. 103-106, 2001.

[17] “DAP 840/103, MSTB 010-06-C1Z Technical Data Manual,” Micro-star Laboratories, Bellevue, WA, 2000.

[18] G.E. Bloom, “IMAGINE II, a converter kit for exploring the fascinating world of integrated-magnetic concepts,”1991.

[19] “PIC16F877 Technical Data Manual,” Microchip, Tucson, Arizona, 2003.

[20] J.-S.R. Jang, C.-T. Sun and E. Mizutani, “Neuro-Fuzzy and Soft Computing: A computational Approach to Learning and Machine Intelligence”, Prentice Hall, Englewood Cliff, NJ, 1997.

APPENDIX A

FUZZY LOGIC PROGRAM

The following is the fuzzy logic program written as a function which is called during the experimental verification

```
function [y]=FuzzyContr_1(e,ce)
%Function fuzzy_DC-DC(), takes two input namaly the values of "e" and  "ce"
%and then gives a crisp fuzzy output as "y".

% Loading in the input values
load Input_Err.dat;
load Input_Cer.dat;

%Input_1 membership sets
L=1; ML=2; MH=3; H=4;

%Input_2 membership sets
ce_L=1; ce_M=2; ce_H=3;

% Output membership sets
ctr_L=2; ctr_M=3; ctr_H=4;  %These numbers correspond to the appropriate duty cycles

% Number of rules
NumberOfRules=12;
Area(NumberOfRules)=0;
Index(NumberOfRules)=0;

% for m=1:NumberOfRules;
%     w(m)=1.0;
%     input(m)=0;
% end

% Declaring the number of membership for both iputs
number1=4; number2=3;

% Reading in the  value ranges for the membership set values from an input file
a =Input_Err(1:number1);
b =Input_Err(number1+1:number1*2);
c =Input_Err((number1*2)+1:number1*3);

d =Input_Cer(1:number2);
ee=Input_Cer(number2+1:number2*2);
f =Input_Cer((number2*2)+1:number2*3);
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      for k=1:number1
         if (e>=a(k))&(e<=b(k))
            member1(k)=(e-a(k))/(b(k)-a(k));
        elseif(e>=b(k))&(e<=c(k))
             member1(k)=(e-c(k))/(b(k)-c(k));
         else
            member1(k)=0;
         end
      end
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      for k=1:number2
         if (ce>=d(k))&(ce<=ee(k))
            member2(k)=(ce-d(k))/(ee(k)-d(k));
        elseif(ce>=ee(k))&(ce<=f(k));
          member2(k)=(ce-f(k))/(ee(k)-f(k));
        else
            member2(k)=0;
```

```
        end
    end
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
% Rules
    Rule1= [H   ce_L   ctr_M];
    Rule2= [H   ce_M   ctr_H];
    Rule3= [H   ce_H   ctr_H];
    Rule4= [MH  ce_L   ctr_L];
    Rule5= [MH  ce_M   ctr_M];
    Rule6= [MH  ce_H   ctr_H];
    Rule7= [ML  ce_L   ctr_L];
    Rule8= [ML  ce_M   ctr_M];
    Rule9= [ML  ce_H   ctr_M];
    Rule10=[L   ce_L   ctr_L];
    Rule11=[L   ce_M   ctr_L];
    Rule12=[L   ce_H   ctr_M];

    Rules=[Rule1;Rule2;Rule3;Rule4;Rule5;Rule6;Rule7;Rule8;Rule9;Rule10;Rule11;Rule12];
 %ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
    sum1=0;
    sum2=0;
    for m=1:NumberOfRules
        if (member1(Rules(m,1)))<=(member2(Rules(m,2)))
           memberout=member1(Rules(m,1));
        else
           memberout=member2(Rules(m,2));
        end
        Area(m)=memberout;
        Index(m)=Rules(m,3);
        sum1=sum1+Area(m)*Index(m);
        sum2=sum2+Area(m);
    end
    sum2=sum2+(sum2==0)*eps;
    CrispOutput=sum1/sum2;
    y=round(CrispOutput);
%vvvvvvvvvvvvvvvv   THE END vvvvvvvvvvvvvvvv
```

The following is the "online" code running in matlab during the experimental verification

```
%% This example writes matrix data to the DAP and reads processed
%% data from the DAP.

close;
clear;

%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
% Open text and binary handles to ACCEL device
texthandle = dapopen('\\.\dap0\$sysin', 'write')
  if texthandle == 0
    error('Error opening DAP text handle')
  end
binaryhandle = dapopen('\\.\dap0\$binout', 'read')
  if binaryhandle == 0
    error('Error opening DAP binary handle')
  end
binputhandle = dapopen('\\.\dap0\$binin', 'write')
  if binputhandle == 0
    error('Error opening DAP $binin handle')
  end

dappstr(texthandle, 'RESET');                % Send a command to reset the DAP
dapflshi(binaryhandle);                      % Flush old DAP data in binary pipe

% Configure DAP using DAPL command file BIWAY.DAP
cnfg = dapcnfig(texthandle, 'dat_acq1.dap');
  if cnfg < 1
    error('Error configuring DAP')
  end
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
w=4;
count =1;
s1 = serial('COM1');
set(s1,'BaudRate',19200);
fopen(s1);
while count<3000;
   newwave = dapgetm(binaryhandle, [1, 1], 'int16');  % Get processed data from DAP
   new_data=newwave/6673 +0.05;

   e=new_data;
   ce=w;
   if e < 0.9
      e=0.9;
   end
   e_saved(count)=e;
   ce_saved(count)=ce;

   %ccccccccccccccccccccccc CALL CONTROLLER ccccccccccccccccccccccccccc
   w=FuzzyContr_1(e,ce)
   w_saved(count)=w;
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
   str = num2str(w);
   fprintf(s1,'%s',str);
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```
  count = count +1;
end
  fclose(s1);
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
% Terminate DAP communication
dappstr(texthandle,'STOP');             % Send a command to stop the DAP
% Close text and binary handles to ACCEL device
textclose = dapclose(texthandle);
binclose = dapclose (binaryhandle);
binputclose = dapclose (binputhandle);
%ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

PIC16F877 PROGRAM FOR DUTY CYCLE GENERATION

```
;SERIAL COMMUNICATION: DUTY CYCLE CALCULATION

list P = 16f877
;Register definitions
STATUS          EQU     0x03
INDF            EQU     0x00
FSR             EQU     0x04
PIR1            EQU     0x0C
PIE1            EQU     0x8C
RCSTA           EQU     0x18
TXREG           EQU     0x19
TXSTA           EQU     0x98
SPBRG           EQU     0x99
RCREG           EQU     0x1A
FSR             EQU     0x04
INDF            EQU     0x00
PORTD           EQU     0x08
TRISD           EQU     0x88
TMRO            EQU     0x01
INTCON          EQU     0x0B
OPTION_REG      EQU     0x81
B1              EQU     0x03

;Data Space
        CBLOCK          0x20    ;RAM AREA for USE at address 20h
        first
        second
        third
        forth
         PWord1
        PWord2
        PWord3
        PWord4
        PWord5
        PWord6
        PWord7
        PWord8
        PWord9
        fifth
        Trial1
        Trial2
        Trial3
        Trial4
        Count
        TimeC
        TimeC1
        TimeC2
        ENDC            ;end of ram block
;The Next 5 lines must be here
;because of bootloader arrangement
;Bootloader first execute the first 4 addresses
;then jump to the address that the execution directs
;=================================================
        ORG     0x0000          ;line 1
        GOTO    START           ;line 2 ($0000)
        NOP                     ;line 3 ($0001)
        NOP                     ;line 4 ($0002)
        NOP                     ;line 5 ($0003)
```

```
;=================================================

;START OF THE PROGRAM FROM $4000
START bsf     STATUS,0x05   ;'from 00 to 01' set RAM Page 1
      movlw   0x00          ;RA0 - RA7 are outputs
      movwf   TRISD
;SETTING THE BAUD RATE TO 19200
      movlw   0x40
      movwf   SPBRG
      bsf     TXSTA,0x02    ;Setting bit BRGH for high speed baud rate
;ENABLING THE ASYNCHROUNOUS SERIAL PORT
      bcf     TXSTA,0x04    ;Clearing the bit SYNC
;ENABLING TRANSMISSION
      bsf     TXSTA,0x05    ;Setting TXEN to enable transmission, this will also set bit TXIF
      bcf     STATUS,0x05   ;Back to Bank 0
      bsf     RCSTA,0x07    ;Setting bit SPEN
;*******STORING OF THE PASSWORD USING INDIRECT ADDRESSING*************
      movlw   0x24
      movwf   FSR
      movlw   31
      movwf   INDF          ;This is to Store "1" at address "24"
      incf    FSR
      movlw   32
      movwf   INDF          ;This is to Store "2" at address "25"
      incf    FSR
      movlw   33
      movwf   INDF          ;This is to Store "3" at address "26"
      incf    FSR
      movlw   34
      movwf   INDF          ;This is to Store "4" at address "27"
      incf    FSR
      movlw   35
      movwf   INDF          ;This is to Store "5" at address "28"
      incf    FSR
      movlw   36
      movwf   INDF          ;This is to Store "6" at address "29"
      incf    FSR
      movlw   37
      movwf   INDF          ;This is to Store "7" at address "2A"
      incf    FSR
      movlw   38
      movwf   INDF          ;This is to Store "8" at address "2B"
      incf    FSR
      movlw   39
      movwf   INDF          ;This is to Store "9" at address "2B"
;*********************************************************************************************

;*********************************************************************************************
      clrf    PORTD
Sind1 movlw   0x01
      NOP
      NOP
      movwf   TimeC
      bsf     RCSTA,0x04    ;Enabling the reception by setting CREN
loop1 bsf     PORTD,B1      ;led on
      decfsz  TimeC
      goto    loop1
      btfss   PIR1,0x05     ;checking hyperterminal if any data has been received
      goto    Sind2
      goto    Chk1
```

```
Sind2   movlw   0x09
        movwf   TimeC
Loop2   bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2
        goto    Sind1
;*****************************************************************************************

;*****************************************************************************************
Chk1    movf    RCREG,0
        movwf   Trial1
        movf    PWord1,0        ;Checking with the first number in Pword1
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk2
        goto    Sind1
Chk2    movf    PWord2,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk3
        goto    Sind1_2
Chk3    movf    PWord3,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk4
        goto    Sind1_3
Chk4    movf    PWord4,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk5
        goto    Sind1_4
Chk5    movf    PWord5,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk6
        goto    Sind1_5
Chk6    movf    PWord6,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk7
        goto    Sind1_6
Chk7    movf    PWord7,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk8
        goto    Sind1_7
Chk8    movf    PWord8,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Chk9
        goto    Sind1_8
Chk9    movf    PWord9,0
        subwf   Trial1,0
        btfss   STATUS,0x02
        goto    Sind1           ;If the no. did not match, go to Sind1
        goto    Sind1_9
;*****************************************************************************************

;*****************************************************************************************
        clrf    PORTD
```

```
Sind1_2 movlw   0x02
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_2 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_2
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_2
        goto    Chk1
Sind2_2 movlw   0x08
        movwf   TimeC
Loop2_2         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_2
        goto    Sind1_2
;*****************************************************************************************

;*****************************************************************************************
        clrf    PORTD
Sind1_3 movlw   0x03
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_3 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_3
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_3
        goto    Chk1
Sind2_3 movlw   0x07
        movwf   TimeC
Loop2_3         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_3
        goto    Sind1_3
;*****************************************************************************************

;*****************************************************************************************
        clrf    PORTD
Sind1_4 movlw   0x04
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_4 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_4
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_4
        goto    Chk1
Sind2_4 movlw   0x06
        movwf   TimeC
Loop2_4         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_4
        goto    Sind1_4
;*****************************************************************************************
```

```
;*****************************************************************************************
        clrf    PORTD
Sind1_5 movlw   0x05
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_5 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_5
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_5
        goto    Chk1
Sind2_5 movlw   0x05
        movwf   TimeC
Loop2_5         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_5
        goto    Sind1_5
;*****************************************************************************************
;*****************************************************************************************
        clrf    PORTD
Sind1_6 movlw   0x06
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_6 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_6
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_6
        goto    Chk1
Sind2_6 movlw   0x04
        movwf   TimeC
Loop2_6         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_6
        goto    Sind1_6
;*****************************************************************************************
;*****************************************************************************************
        clrf    PORTD
Sind1_7 movlw   0x07
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_7 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_7
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_7
        goto    Chk1
Sind2_7 movlw   0x03
        movwf   TimeC
Loop2_7         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_7
        goto    Sind1_7
;*****************************************************************************************
```

```
;*****************************************************************************************
        clrf    PORTD
Sind1_8 movlw   0x08
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_8 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_8
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_8
        goto    Chk1
Sind2_8 movlw   0x02
        movwf   TimeC
Loop2_8         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_8
        goto    Sind1_8
;*****************************************************************************************
;*****************************************************************************************
        clrf    PORTD
Sind1_9 movlw   0x09
        NOP
        NOP
        movwf   TimeC
        bsf     RCSTA,0x04      ;Enabling the reception by setting CREN
loop1_9 bsf     PORTD,B1        ;led on
        decfsz  TimeC
        goto    loop1_9
        btfss   PIR1,0x05       ;checking hyperterminal if any data has been received
        goto    Sind2_9
        goto    Chk1
Sind2_9 movlw   0x01
        movwf   TimeC
Loop2_9         bcf     PORTD,B1        ;led off
        decfsz  TimeC
        goto    Loop2_9
        goto    Sind1_9
;*****************************************************************************************
        End                     ;End of the program
;*************************************ends*************************************************
```